Generating Asserts for Test Cases Efficiently

Student: Luís Pinho

Supervisor: Rui Maranhão (PhD)
Co-Supervisor: José Campos (MSc)

Universidade do Porto
FEUP Faculdade de Engenharia

10th July, 2013

Agenda

- Context
- Motivation
- State of the art
- LEOPArD
- Evaluation
- Conclusions
- Future Work
- Contributions

Context

Context


test
intended
number
include
Institute
tools
separate
methods
time
agile
expected
also
example
oriented
executed
failure
following
may
analysis
black
Compatibility
citation
team
system
Assurance
code
whether
actual
process
design
development
steps
output
regression
plan
source
acceptance
set
defects
defect
used
often
requirements
beta
non-functional
tester
box
quality
product
developers
value
vs
integrated
one
end
white
Input
exist
tested
failures
actually
fault
different
offered
conditions
well
information
needed
Certification
unit
functionality
considered
Certifications
case
program
testers
tests
two
faults
alpha
inputs
reports
based
cases
usually
result
data
functions
bugs
automated
project
common
results
International
SQA
latest
level
coverage
performance
found

Motivation

Motivation

Therac-25

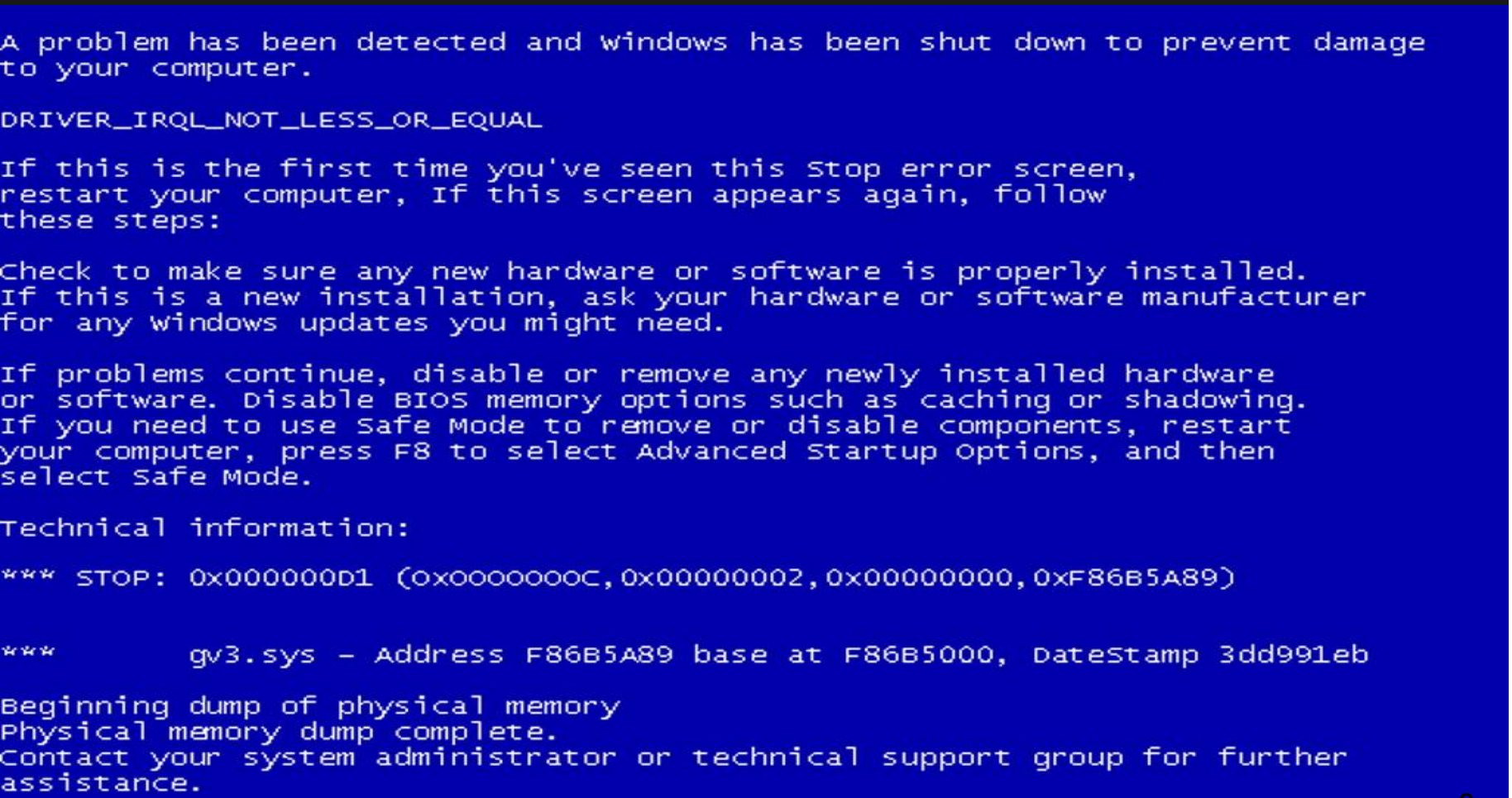
- At least 5 patients died, while several others became seriously injured.
- A race-condition caused the machine to give approximately 100 times the intended dose, causing radiation overdose.

Motivation

The explosion of Ariane 5:

- The explosion occurred after 37 seconds of start (30 seconds after lift-off).
- The exception was caused during execution of a data conversion from 64-bit floating point to 16-bit signed Integer value.

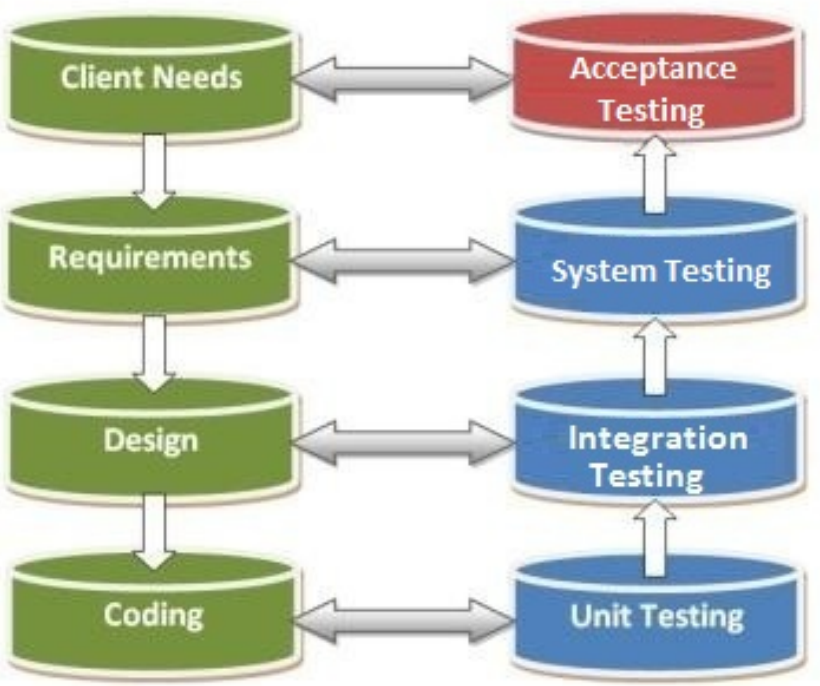
Motivation


A problem has been detected and Windows has been shut down to prevent damage
to your computer.
DRIVER_IRQL_NOT_LESS_OR_EQUAL
If this is the first time you've seen this Stop error screen,
restart your computer, If this screen appears again, follow
these steps:
Check to make sure any new hardware or software is properly installed.
If this is a new installation, ask your hardware or software manufacturer
for any Windows updates you might need.
If problems continue, disable or remove any newly installed hardware
or software. Disable BIOS memory options such as caching or shadowing.
If you need to use Safe Mode to remove or disable components, restart
your computer, press F8 to select Advanced Startup Options, and then
select Safe Mode.
Technical information:
*** STOP: 0x000000D1 (0x0000000C,0x00000002,0x00000000,0xF86B5A89)
*** gv3.sys - Address F86B5A89 base at F86B5000, DateStamp 3dd991eb
Beginning dump of physical memory
Physical memory dump complete.
Contact your system administrator or technical support group for further
assistance.

Motivation

Fully Automated Test Case Generation

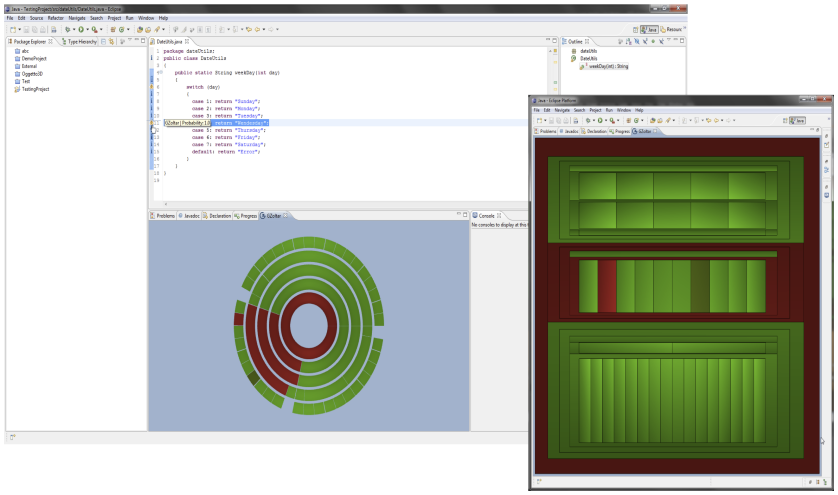
Software Development Cycle



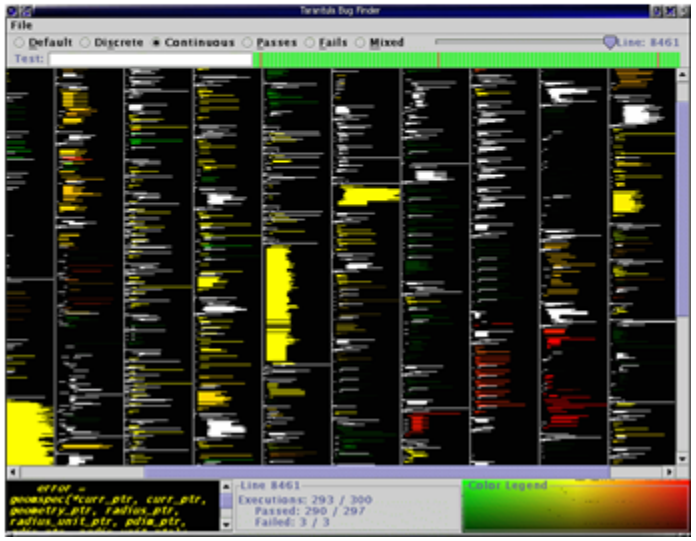
The Debug Phase

Automatic Debugging Tools

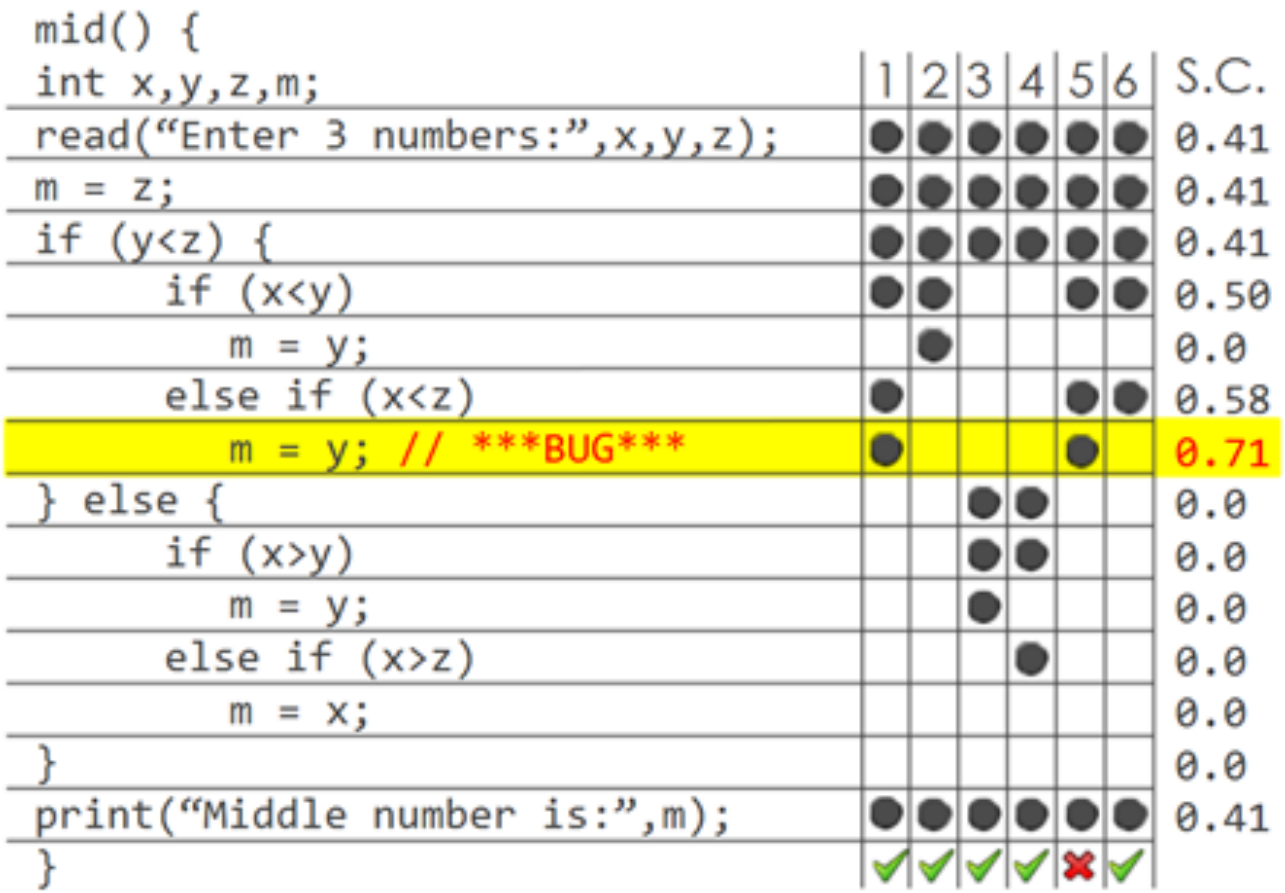
GZoltar



Tarantula



Spectrum-based fault localization (SFL)



Fault screeners (Invariants)

An Invariant is a data structure that is valid during the execution of a program.

Range Invariant

Stores the the values observed in a **range** structure.

$10 < x < 20$
("x" is valid between 10 and 20)

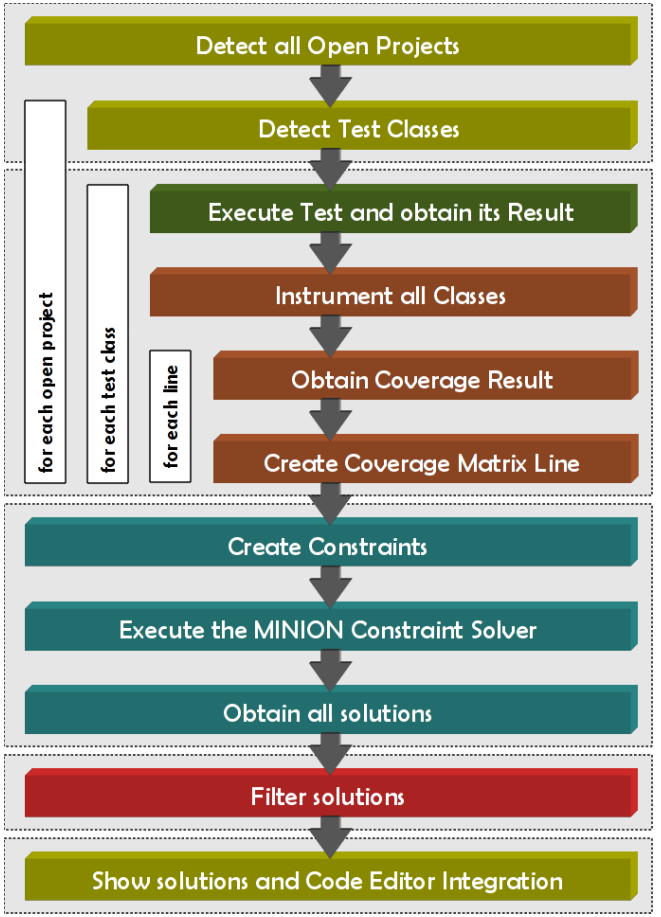
Bloom Filters

Stores the values observed in a **set** structure.

$x = \{10, 20, 30\}$
("x" is valid for the values 10, 20 and 30)

GZoltar

GZOLTAR



Goal

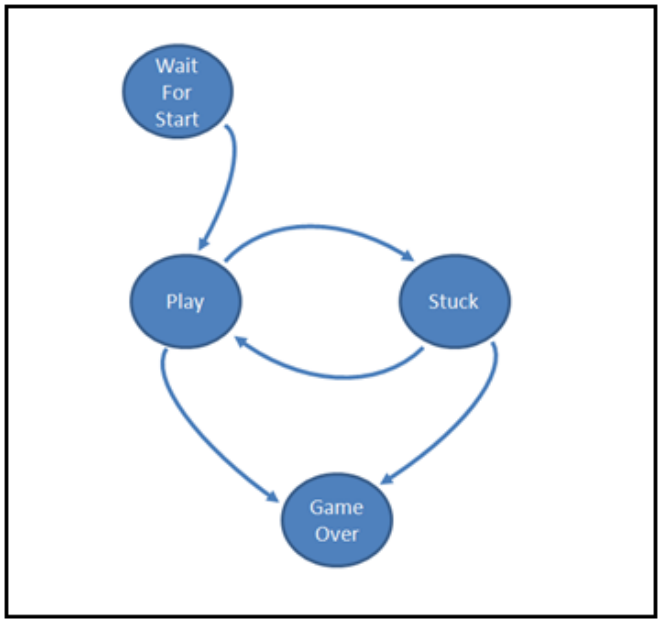
Main Goal

```
public class Add extends TestCase
{
	@Test
	public void testAdd() {
	int result = Calculator.add(1,2);

	assertTrue(result == 3);
	}
}
```

How?

Dynamically Inferring Behavioural Models with Fault Screeners

LEOPArD



Training Phase

- LEOPArD learns the expected behaviour of a software program using correct executions in a so-called training phase.
- The Finite State Machine is created.
- The Invariants are initialized with primary types.

Operational Phase

- After the model is learnt and all invariants are trained, each execution during the operational phase is checked against it for consistency.

- LEOPArD uses the behavioural model learnt to act as an oracle to decide the pass/fail of an execution.

First step - LEOPArD

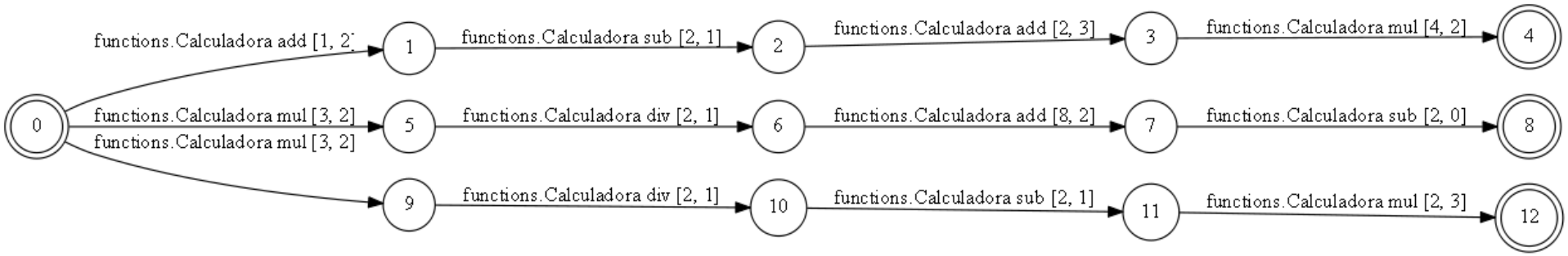
```
TEST 1
add(1,2)     ->     sub(2,1)     ->     add(2,3)     ->     mult(4,2)
   =3                  =2                  =5                  =8

TEST 2
mult(3,2)    ->     div(2,1)     ->     add(8,2)     ->     sub(2,0)
   =6                  =2                  =10                 =2

TEST 3
mult(3,2)    ->     div(2,1)     ->     sub(2,1)     ->     mult(2,3)
   =6                  =2                  =1                  =6

TEST 4
add(2,3)     ->     sub(3,2)     ->     add(3,4)     ->     mult(5,3)
   =5                  =1                  =7                  =15
```

Second step - LEOPArD

```
TEST 1
add(1,2)     ->    sub(2,1)     ->    add(2,3)     ->    mult(4,2)
   =3                 =2                 =5                 =8

TEST 2
mult(3,2)    ->    div(2,1)     ->    add(8,2)     ->    sub(2,0)
   =6                 =2                 =10                =2

TEST 3
mult(3,2)    ->    div(2,1)     ->    sub(2,1)     ->    mult(2,3)
   =6                 =2                 =1                 =6

TEST 4
add(2,3)     ->    sub(3,2)     ->    add(3,4)     ->    mult(5,3)
   =5                 =1                 =7                 =15
```

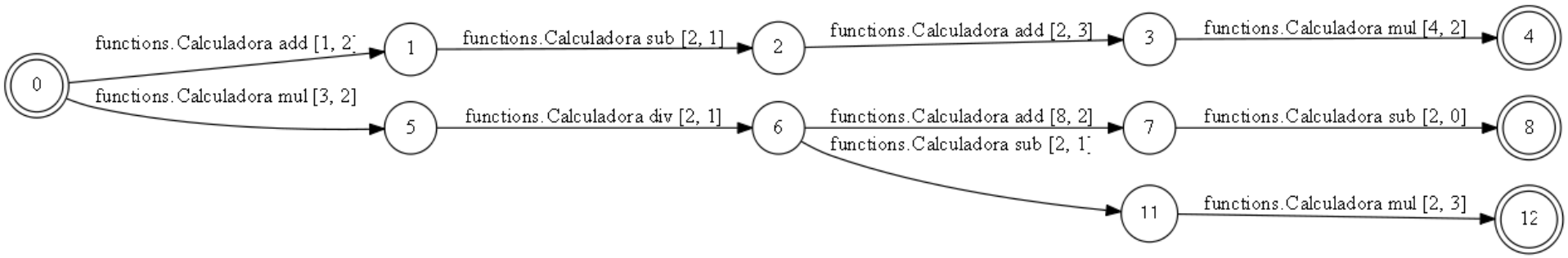
Third step - LEOPArD

State Machine



Third step (+1) - LEOPArD

State Machine + Equivalence (k=2)

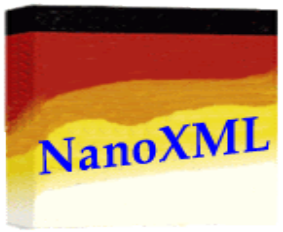


Evaluation

Evaluation (Last Step) - LEOPArD

```
P <- INJECT FAULT IN P
for x in {10,30,50,70,90,100} do
    for i in [1..20] do
        T' <- SELECT x passing test cases from T
        FSM_B <- LEARN (P,T')
        if OperationalPhase then
            ResultsTEMP += CollectResultsFromP (Fp, Fn, SFL)
        end if
    end for
    Results += AVERAGE (ResultsTEMP)
end for
```

Experimental Setup



Subject	Version	LOCs	Test Cases	Coverage	Number of States
ATM	0.1	2418	20	14.8%	61
NanoXML	2.2.3	5393	9	45.7%	6050
org.jacoco.report	0.5.7	5979	99	73.1%	763

- Single Bug (3)

- Double Bug (3)

- Triple Bug (1)

Experimental Setup

- Check the following cases:
  - The sequence doesn't exist -> FAIL
  - The sequence exists but the ranges were violated -> FAIL
  - The sequence exists and the ranges were accepted -> PASS

Evaluation Metrics

- False Positive Rate

$$f_p := \frac{N_{fp}}{N_t}$$

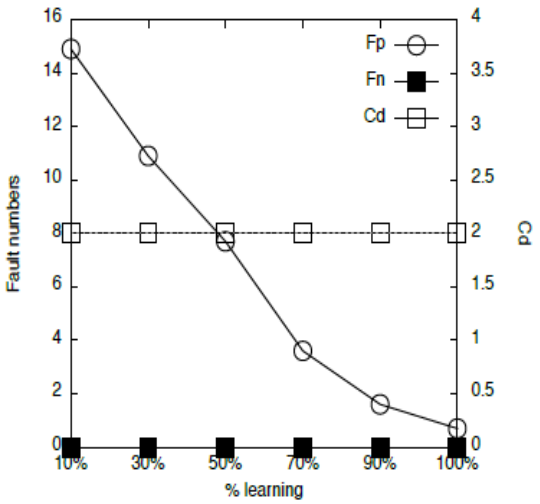
- False Negative Rate

$$f_n := \frac{N_{fn}}{N_t}$$

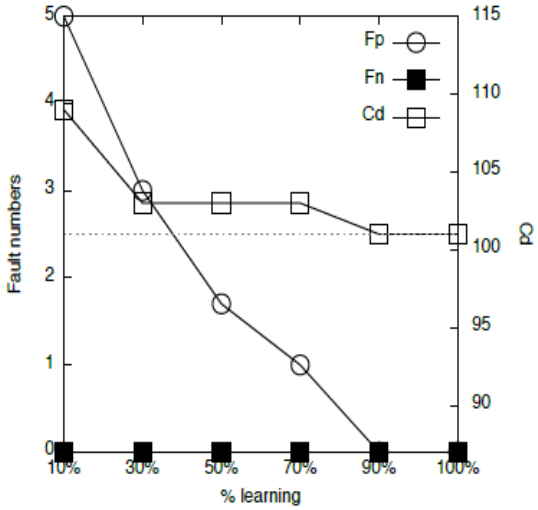
- Cost of diagnosis

$$C_d = \frac{|\{i|s_O(i) > s_O(d*)\}| + |\{i|s_O(i) \geq s_O(d*)\}| - 1}{2}, \; 1 \leq i \leq M$$

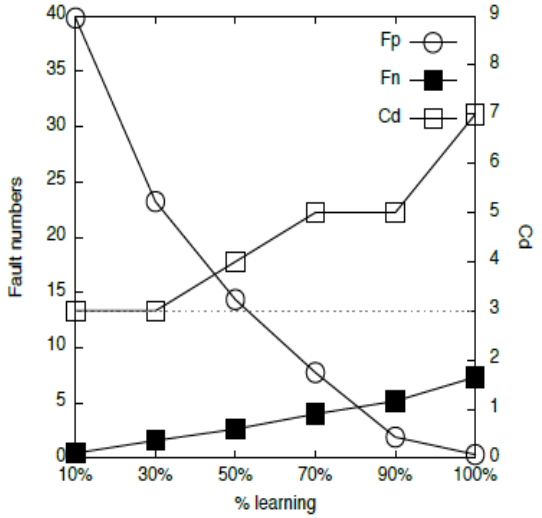
Results - Single Bug



(a) ATM single fault.

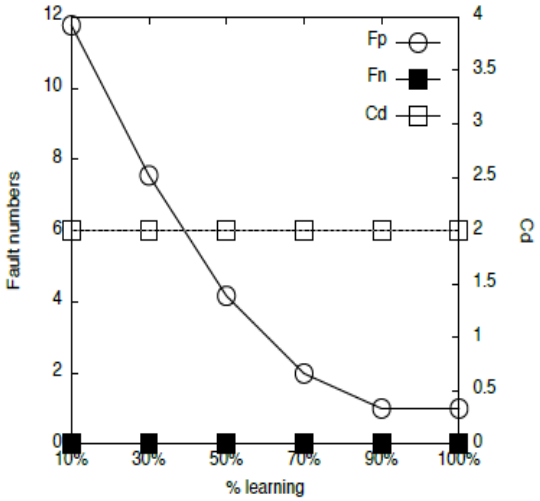


(b) NanoXML single fault.

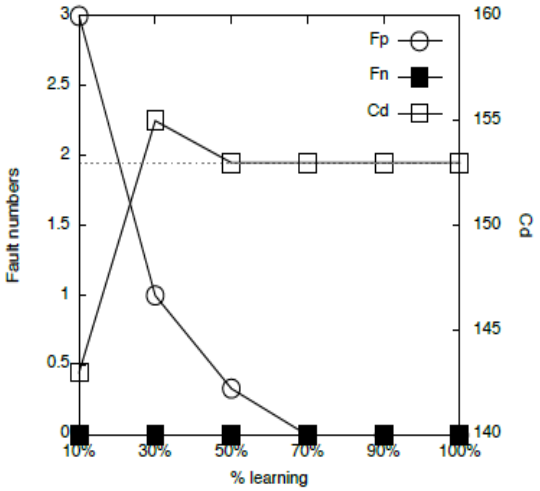


(c) org.jacoco.report single fault.

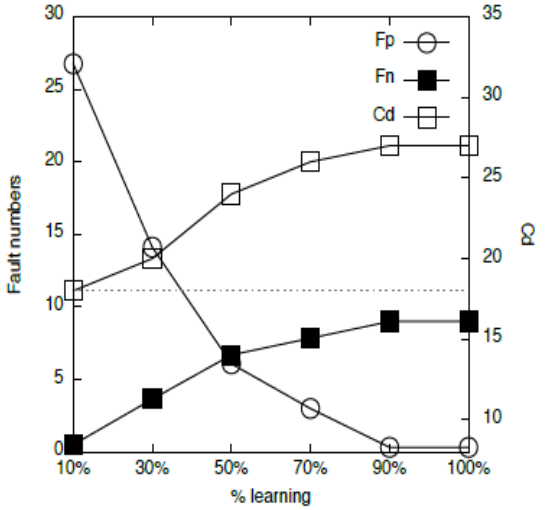
Results - Double Bug



(a) ATM double fault.

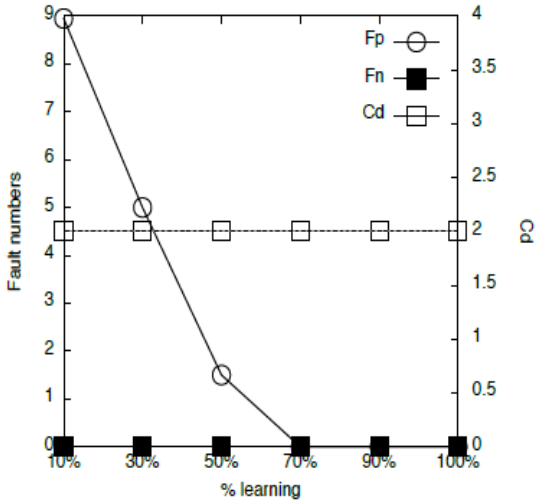


(b) NanoXML double fault.

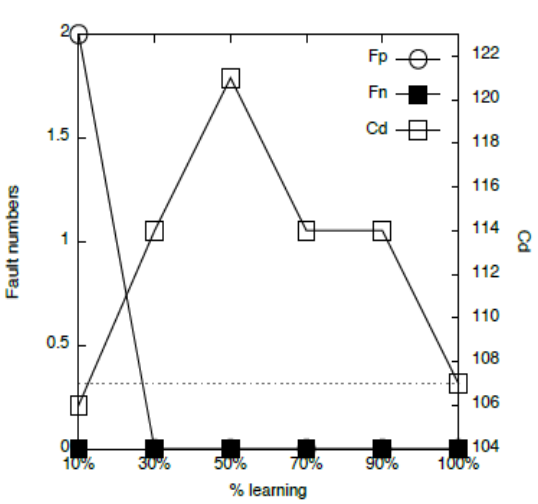


(c) org.jacoco.report double fault.

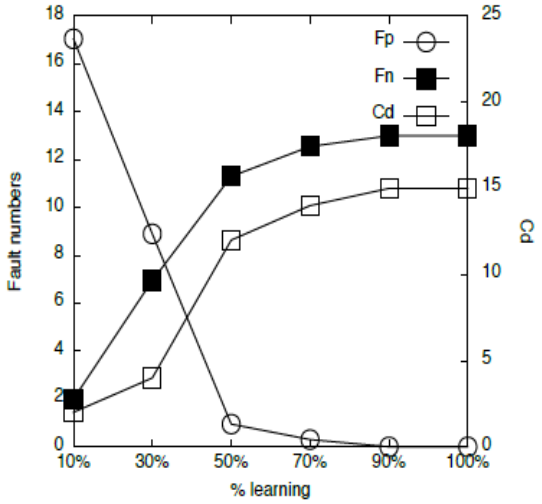
Results - Triple Bug



(a) ATM triple fault.



(b) NanoXML triple fault.



(c) `org.jacoco.report` triple fault.

Conclusions

Conclusions

- LEOPArD yields similar diagnostic quality, while entailing acceptable runtime overhead.

- With only 30% of the total passing test cases, we can obtain similar diagnostic accuracy (only 2 more statements that need inspection), despite entailing an average of 8 false positives and 1 false negative.

Future Work

- Monitoring of non-primitive variables. (String, char...)
- Tune the behavioural model in order to decrease the slowdown
- Algorithms to reduce the state machine size

Contributions

- **IJUP'13** <- Accepted
- **ICTSS'13** <- Submitted and under review

Discussion

ZOLTAR

AutoSeer